CS534 - Machine Learning

Implementation Assignment 4

Group 50
Sebastian Mueller - 933962290
Derek Helms - 934451909
Vy Bui - 934370552

Instructor: Dr. Xiaoli Fern

The School of Electrical Engineering and Computer Science
Oregon State University
Nov 22, 2022

Word Embedding Exploration

Question 1:
list the 29 most similar words for each seed word.

Response:

rank	word	rank	word	rank	word	rank	word	rank	word
1	plane	1	great	1	horrible	1	need	1	early
2	flights	2	well	2	awful	2	helping	2	earlier
3	boarding	3	nice	3	bad	3	please	3	usual
4	airline	4	better	4	brutal	4	pls	4	after
5	jet	5	night	5	idea	5	let	5	again
6	flying	6	bad	6	horrendous	6	us	6	saturday
7	heading	7	morning	7	horrid	7	give	7	afternoon
8	arrival	8	way	8	shitty	8	trying	8	hour
9	airlines	9	hope	9	quite	9	can	9	guess
10	travel	10	but	10	worst	10	helps	10	missed
11	shuttle	11	too	11	similar	11	must	11	work
12	delayed	12	really	12	shame	12	tell	12	hours
13	landing	13	right	13	worse	13	find	13	sunday
14	route	14	through	14	crap	14	could	14	since
15	airplane	15	there	15	actual	15	plz	15	night
16	safe	16	day	16	horrific	16	helped	16	away
17	booking	17	luck	17	bloody	17	support	17	yesterday
18	fly	18	sure	18	ridiculous	18	anyone	18	last
19	departure	19	it	19	such	19	should	19	maybe
20	waiting	20	thing	20	atrocious	20	save	20	yet
21	landed	21	pretty	21	dreadful	21	take	21	monday
22	journey	22	think	22	sick	22	want	22	wait
23	passengers	23	have	23	wtf	23	bring	23	either
24	transit	24	all	24	fucking	24	maybe	24	mins
25	delay	25	yes	25	cruel	25	lets	25	wake
26	crew	26	very	26	seriously	26	seriously	26	before
27	pilot	27	again	27	unreal	27	able	27	thursday
28	trip	28	work	28	mess	28	here	28	hopefully
29	taxi	29	yeah	29	however	29	needs	29	friday

Table 1: Tables from left to right display 29 most similar words to words "flight", "good", "terrible", "help", and "late" respectively.

Question 2:
Do you observe five distinct clusters in your visualization?

Response:

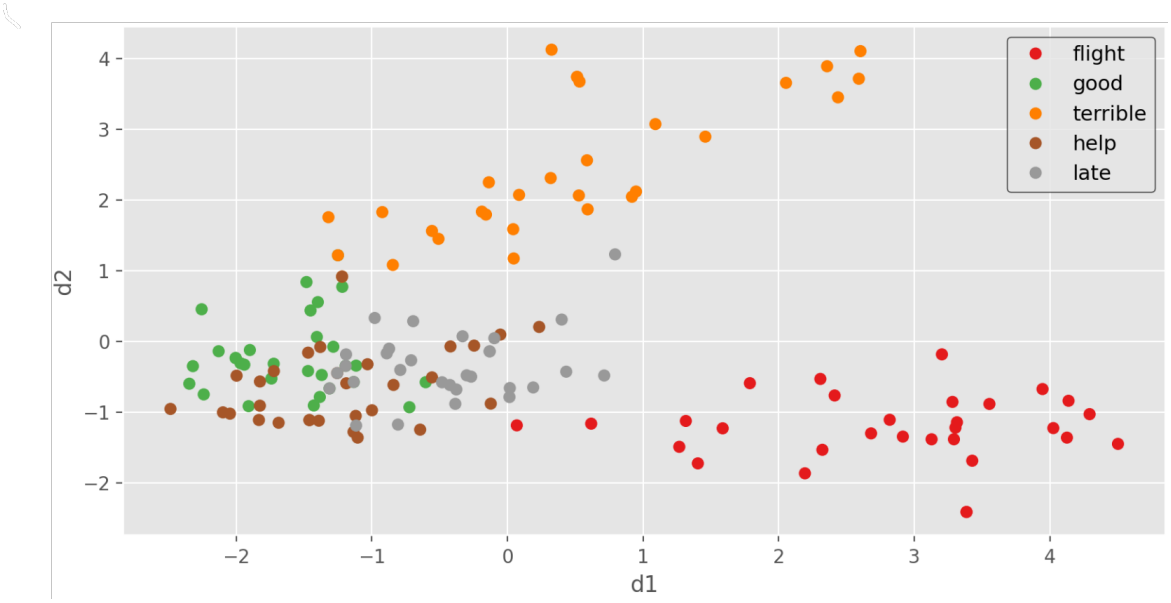

Figure 1: PCA on 150 Clustered Words

We did not get five distinct clusters (Figure 1). There is an overlap between the words closest to "good", "help", "late". The words closest to "flight" and "terrible" have a clearer separation in the clusters.

Question 3:
Provide substantially different visualization results by using different perplexity parameter for t-SNE .

Response:
Figures 2 & 3 showcase our exploration of different perplexity values. You can dramatic shifts in clustering tightness and location, even between successive increments sometimes. Such as in perplexity = 41 and perplexity = 43 in Figure 3. These changes could be caused by different initialization since t-SNE has non convex cost functions, which means it can get stuck in different local minimum.

Figure 2: t-SNE exploration for perplexity values 3 - 17

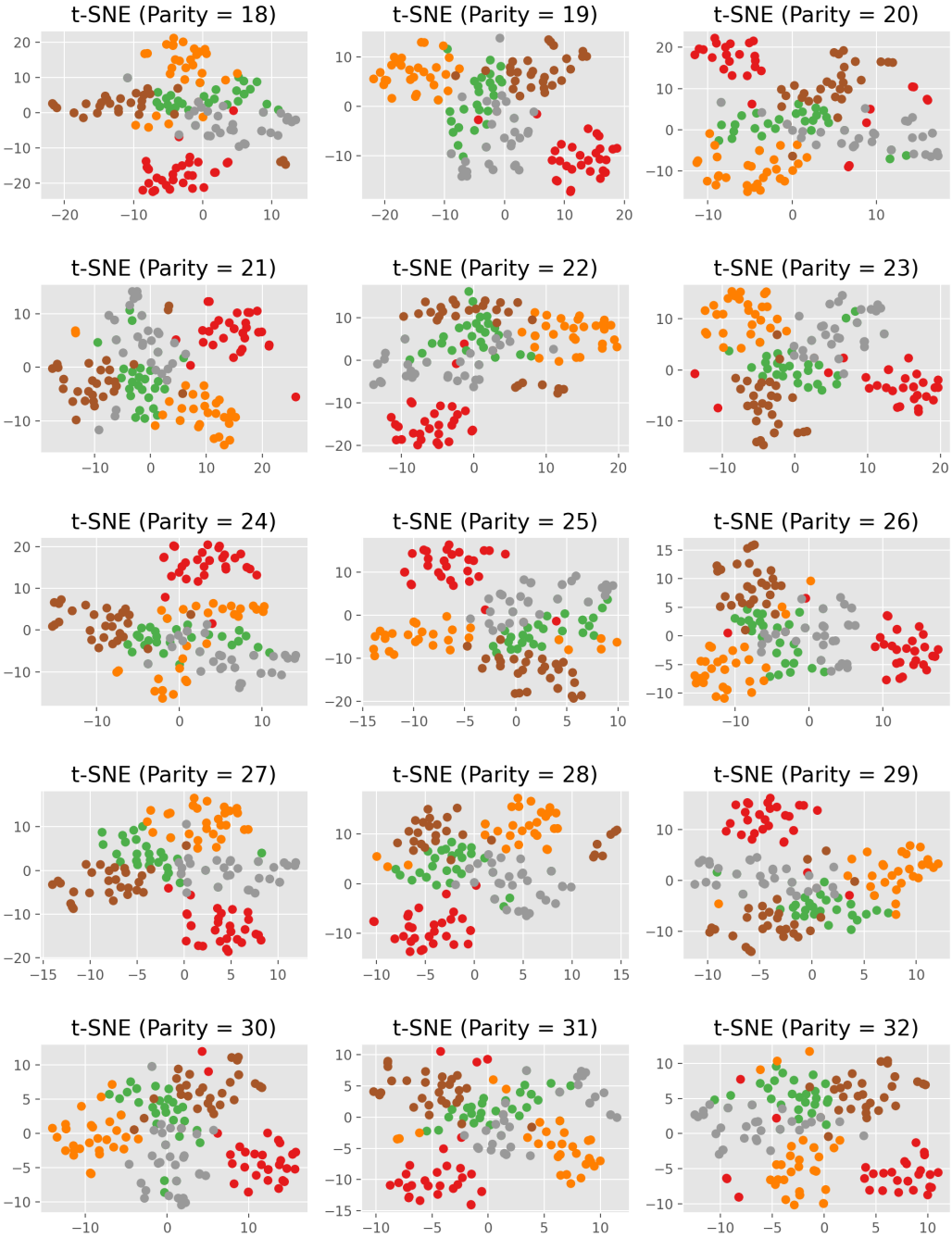

Figure 3: t-SNE exploration for perplexity values 18 - 32

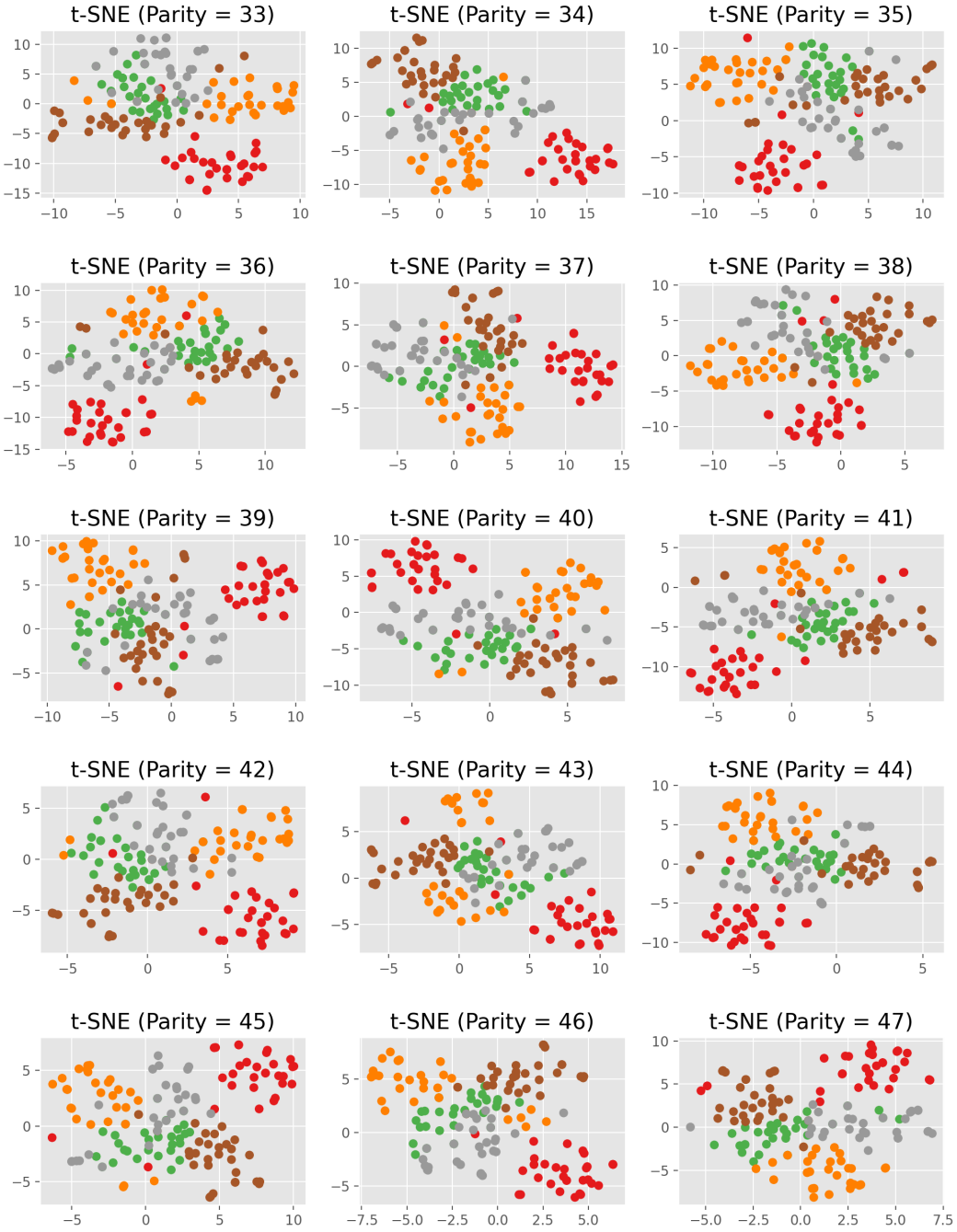

Figure 4: t-SNE exploration for perplexity values 33 - 47

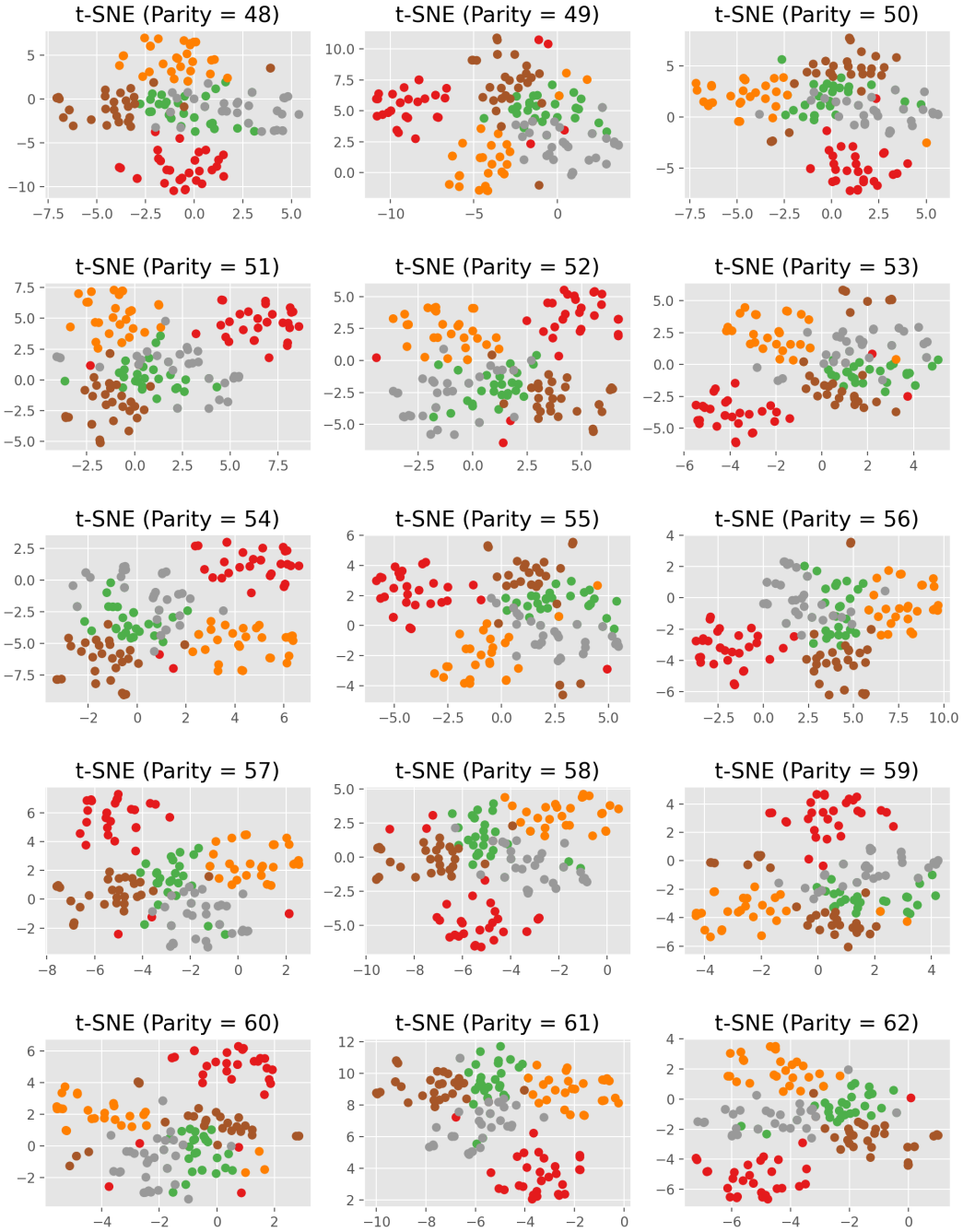

Figure 5: t-SNE exploration for perplexity values 48 - 62

Question 4 (clustering):
Plot the kmeans objective as a function of k. Do you observe monotonically decreasing objective value as we increase k? Do you see any evidence from this curve that suggests k = 5? Provide an explanation for your results.

Response:
We do observe a monotonically decreasing objective value as we increase k. We would want to look for hitches or elbow points in this graph to use as evidence for optimal k value. Looking at Figure 6 we see that there are no blatantly standout sharp elbow points – so we may value multiple elbow points equally with this inspection method.

Increasing the scale of Figure 6 (Figure 7), we can see there is an elbow point at $k \approx 5$. This may provide evidence of to suggest k=5, however like mentioned above it may be hard to pick a "best elbow" point given a lack of clear standout elbow point. Consequentially, we would want to use additional methods to determine if k=5 is a good choice.

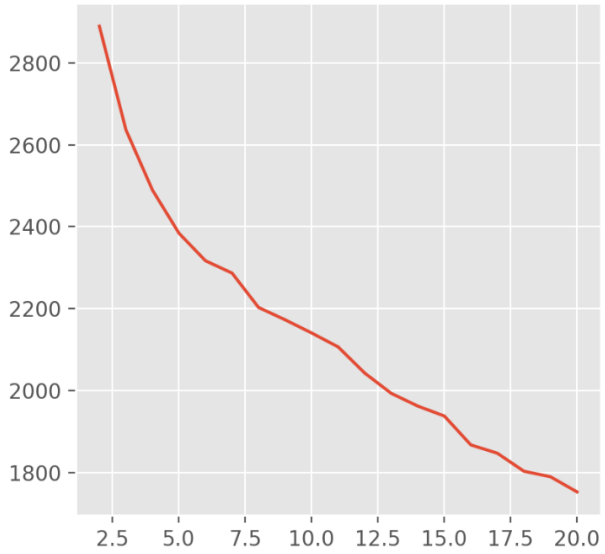

Figure 6: kmeans objective as function of k

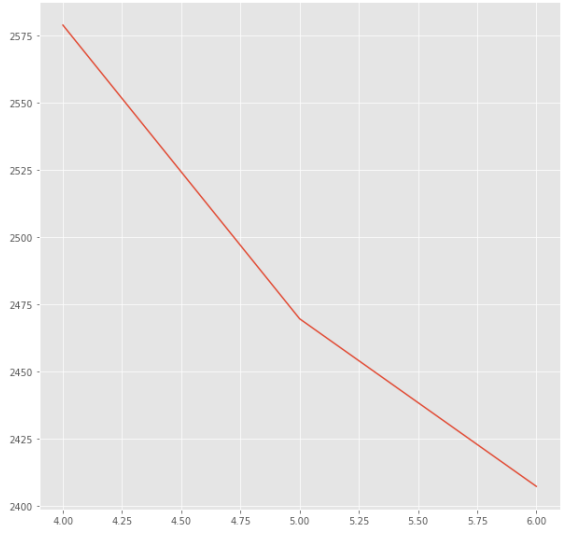

Figure 7: Increased scale of Figure 2 around point of interest for visual inspection

Question 5 (clustering):
Plot each metric you get as a function of k. Do you k = 5 gives the best score for different metrics? Provide an explanation for your observation. Which of these three metrics are appropriate to use if we are evaluating two different clustering algorithms that automatically search for the number of clusters in the data (that is, one algorithm might find five clusters in the data while the other might find ten)?

Response:

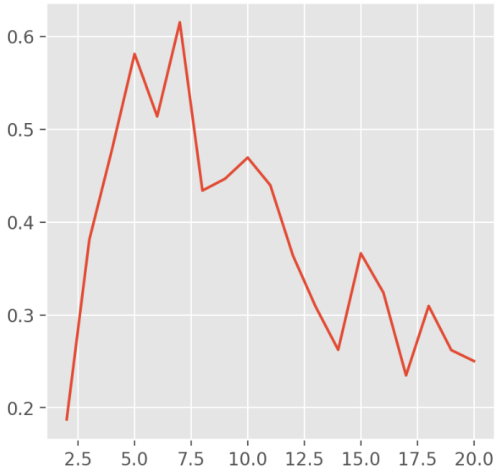

Figure 8: Adjusted RAND score

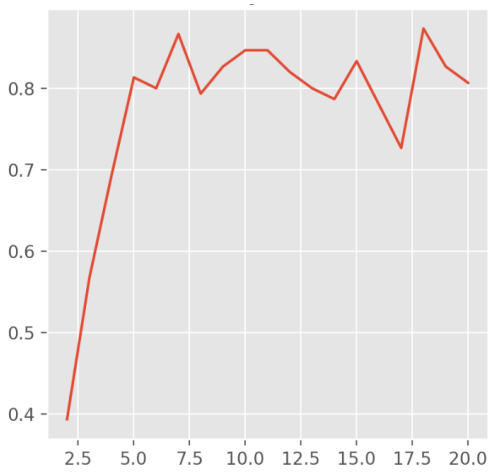

Figure 9: Purity score

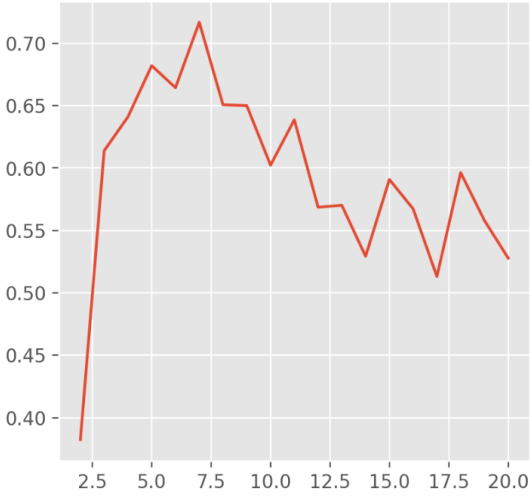

Figure 10: Normalized Mutual information score.

Figures 8-10 showcases our exploration using RAND, mutual information and purity as functions of k.

Thinking about purity, we think k=5 gives us fairly good score; however it looks like 7 could potentially be better. We can think of purity s a measure of the extent that a cluster contains a single class. It is important to note with purity that you can maximize the value by creating the same amount of clusters as there is data. So we must be aware of the amount of data we are evaluating when using this metric.

With the Adjusted RAND we are looking for higher scores. RAND is a metric of similarity, measuring the similarity of two sets of clusters. It determines how similar the elements are. So again K=5 is good but since k=7 is higher, it might be the better choice when viewing this metric. For example 7 clusters may be able to isolate a few more clusters of higher similarity in messy overlap regions.

Normalized Mutual information is similar to RAND in the sense that it is looking at the the similarity between different cluster sets, it just uses slightly different theory. It the reduction of the entropy class labels. Its essentially telling us how the uncertainty about a given class label may decrease when we know the cluster label. Again a higher mutual information score usually is a indication of better performance, as it correlates to low uncertainty. So K=7 may be a better choice than k=5 in when using this metric.

So in general we would probably decide to further explore k=7 since it preforms the best out of those metrics.

We would want to use adjusted RAND and normalized mutual information since those methods are useful for comparing models with different k values, since the metrics do not assume any type of clustering structure. But we do need some understand of ground truth labels, so if we do not have those we would maybe need to use elbow method or some other type of metric like silhouette method

Word Embedding Exploration

Question 1:
How can you improve the bag-of-words representation for classification using the word embedding?

Response: For our first exploration we explored using weighted averages. Figure 11 highlights our accuracy for linear SVM using this improvement method. We chose to explore this method because we care about sentences, so it makes sense to add all the embeddings together and then "normalize" it with the average to give us information about the sentence itself. Averaging is important since we want to make sure long sentences are not getting larger weightings just due to length solely. This idea of taking average weights works well for identify similar tweets due to the high dimensionality of the embedding space. A convenient outcome of this high dimensionality is that it is extremely unlikely for two different tweets do have the same average (high dimension nearly always produces orthogonal vectors when randomly chosen), while it is guaranteed two similar tweets will have close average embedding weights. This is the logic behind why average weight is a use full tool

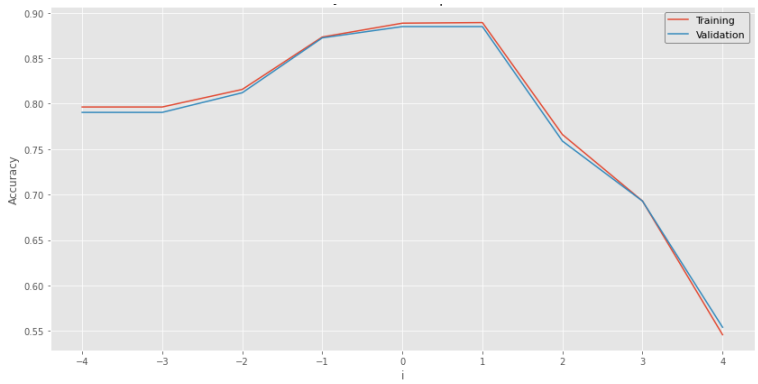

Figure 11: Linear SVM training and validation accuracy after exploring weighted average word embedding

For our second exploration we tried to reduce the noise of each tweet by removing words in the tweet with "@". Usually they were words like "@UnitedAir" or "@SouthwestAir". We did not see any significant change in our training and validation accuracy. This told us that these words were not significant sources of noise.

The third approach we took we sort of thought as a "semi-supervised" approach. With the GloVe embedder there are words in our tweets that are not vocabulary in the GloVe embedder – so we may loose valuable information when those words are assigned 0. Our idea was to overcome the issue by using another vector representation, we used TFID vectorize. We re-vecotrized all the words in our tweets and then used kmeans to cluster the data. After we clustered the data we then calculated the weighted average embedding of those clusters using GloVe embedder. We left out the zero vocabulary not in GloVe when calculating those weighted averages. Once we determined the weighted average of a cluster from the kmeans we implemented, we then assigned those weighted averages to the words that had an initial zero value in from GloVe. We could then work with all word receiving some kind of value.

Unfortunately, we could not get this idea to fully work properly to produce an accuracy graph, but we explored the idea and code thoroughly thinking about this idea and implementation.

General Discussion/Reflection

This was one of the most applied projects we were tasked with this semester as a team, mostly because it was the most open ended without many "guided" instructions or questions in part 2. This simulates a real work environment. Our biggest takeaway away – it is challenging and sometimes frustrating. We came across various ideas we thought would be great to then realize that they did not improve performance at all. We then had to go back to the drawing board. It was a good test of our resilience as problems solvers we've been working on this term.

Talking more specifically to the results of this assignment. We found it interesting that removing tweets with "@" did not reduce the noise in our data. Additionally we were not initially expecting so much variety in our t-SNE perplexity graphs, but then realized how we can arrive at different local minimum.